Biodiversity Informatics, 3, 2006, pp. 1-15

TAXONOMIC NAMES, METADATA, AND THE SEMANTIC WEB

RODERIC D. M. PAGE
Division of Environmental and Evolutional Biology
Institute of Biomedical and Life Sciences
University of Glasgow, Glasgow G12 8QQ, Scotland
Email: r.page@bio.gla.ac.uk

Abstract.— Life Science Identifiers (LSIDs) offer an attractive solution to the problem of globally unique identifiers for digital objects in biology. However, I suggest that in the context of taxonomic names, the most compelling benefit of adopting these identifiers comes from the metadata associated with each LSID. By using existing vocabularies wherever possible, and using a simple vocabulary for taxonomy-specific concepts we can quickly capture the essential information about a taxonomic name in the Resource Description Framework (RDF) format. This opens up the prospect of using technologies developed for the Semantic Web to add "taxonomic intelligence" to biodiversity databases. This essay explores some of these ideas in the context of providing a taxonomic framework for the phylogenetic database TreeBASE.

Key words.— Life Science Identifiers, metadata, taxonomic names, Semantic Web, RDF, triple stores.

Integrating diverse sources of digital information is a major challenge facing biodiversity informatics. Not only are we faced with numerous, disparate data providers, each with their own specific user communities, but the information in which we are interested is diverse, and includes taxonomic names and concepts, specimens in museum collections, scientific publications, genomic and phenotypic data, and images. Of course, this problem is not unique to biodiversity informatics — the wider bioinformatics community is keenly aware of this problem (Stein 2003) and indeed it is major topic of discussion concerning the future direction of the World Wide Web ("Web 2.0").

My goal in this paper is to sketch some ideas on how we could create the infrastructure for constructing a distributed system for querying information on biodiversity. My contention is that, thanks to efforts by the Semantic Web community[1] the elements we need are mostly already in place. The two key technologies I will advocate are the Resource Description Format (RDF[2]) developed by the W3C, and the Life Science Identifier (LSID) technology developed by IBM[3]. It is easy to enthuse about a technology and contribute to the "hype" that surrounds it, so I will try and keep my feet on the ground by providing some background on the problem that lead me to this conclusion, and by presenting working implementations wherever possible.

Motivation

Reconstructing the history of life on Earth (the "Tree of Life") is the holy grail of phylogenetics, yet we lack a comprehensive phylogenetic database that stores our efforts at reconstructing this tree. The most comprehensive phylogenetic database we currently have is TreeBASE[4] (Piel et al. 2002). As I've outlined elsewhere (Page 2004) a major limitation of this database is that it has no taxonomic "intelligence." Taxonomic names are entered into TreeBASE without being validated against any external database of names, hence many of the names are not proper scientific names. Efforts to map names in TreeBASE to external databases rapidly run into problems. Around half the names in TreeBASE do not have an exact match in the NCBI's Taxonomy database. Using data cleaning tools (Herbert et al. 2004) or a combination of approximate string matching, regular expressions, and manual matching[5] can improve on this, but a significant fraction of names in TreeBASE still have no obvious counterpart in the NCBI's database. In some cases this is because no DNA

[1] http://www.w3.org/2001/sw/.
[2] http://www.w3.org/RDF/.
[3] http://lsid.sourceforge.net/.
[4] http://www.treebase.org/.
[5] http://darwin.zoology.gla.ac.uk/~rpage/TreeBASE.

sequences have been (or indeed, can be) obtained from those taxa, in which case those names will not be in the NCBI database and hence matches may be sought in other taxonomic databases, such as the Integrated Taxonomic Information System (ITIS[6]), the International Plant Names Index (IPNI[7]), IndexFungorum[8], and the Universal Biological Indexer and Organizer (uBio[9]). Whereas it is relatively easy to search NCBI's Taxonomy because the entire database can be downloaded, this is not the case for most other taxonomic databases.

A Taxonomic Search Engine

In 2004 I started to map TreeBASE names onto various taxonomic databases (results can be viewed[10]). Querying these source manually using their web interfaces is slow and tedious, so I developed a simple federated search engine that queries multiple taxonomic databases for information about a name (Page 2005). The Taxonomic Search Engine supports two basic queries, NameSearch and GetDataForID. The first query (NameSearch) searches a database for a name, and if the name is found returns the name and its identifier in that database. The second query (GetDataForID) "drills down" to get details about a single record in the source database.

Leaving aside the technical details of talking to databases that support very different query interfaces, I had two problems to deal with. The first was how to generate unique identifiers for names from each database. Given that most of the databases use integers as their primary keys, in many cases the same identifier will be used by different databases. As one of many possible examples, 101593 is the identifier for Odonata in NCBI's GenBank and *Dahlia australis* in ITIS. Hence, if I were to store just the identifier it would not be clear what name that identifier referred to. An obvious solution is the idea of a "namespace" that specifies the context for a given identifier. In this case, the identifier from NCBI could be distinguished from that in ITIS by adding prefixes corresponding to the domain name address of the two databases, i.e., adding "ncbi.nlm.nih.gov" and "itis.usda.gov" to the respective identifiers.

The second problem is how to return information about a specific record in a database (e.g., the name, any synonyms, etc.). Given that each database has its own format for returning information (ranging from delimited text, HTML, XML, and SOAP data structures), I transformed the result returned by each database into a common XML format that in turn could be transformed into HTML output for display in a web browser.

So, to facilitate mapping names in TreeBASE onto names in external databases we need (1) a mechanism for generating globally unique identifiers, and (2) standard format for providing information about the object the identifier refers to. Before introducing one possible solution, let us first consider why names themselves are not enough.

Why Taxonomic Names Aren't Enough

The taxonomic name of an organism is a key link between different databases that store information on that organism. However, taxonomic names themselves have serious limitations as identifiers in databases (Kennedy 2003; Kennedy et al. 2005) due to the existence of multiple names (synonyms) for the same taxon, and the use of the same name to refer to different taxa. For example, the genus *Morus* applies to both an animal (the gannet) and a plant (the mulberry tree). Even species names can be identical — a species of wasp and a species of conifer both share the name *Agathis montana*. Furthermore, there may be multiple names for the same taxon. Hence, using names alone to link different data sources can be prone to error. As an example, at the time of writing NCBI's LinkOut feature mistakenly links the catfish genus *Loricaria* (NCBI tax_id = 52085) to the TreeBASE taxon *Loricaria* (TreeBASE TaxonID = 1305), which is a plant genus (family Compositae). This lack of uniqueness of names raises the issue of how to store taxonomic information in databases.

URIs, URLs, and URNs

Life Science Identifiers (LSID) are one solution to the problem of globally unique identifiers (Clark et al. 2004). At the risk of drowning the reader in alphabet soup, it is useful to distinguish between two different types of identifiers in use in the Internet, the Uniform Resource Locator (URL), and the Uniform Resource Name (URN). URNs and URLs are two possible kinds of Uniform Resource Identifier (URI).

[6] http://www.itis.usda.gov/.
[7] http://www.ipni.org/.
[8] http://www.indexfungorum.org/.
[9] http://wwww.ubio.org/.
[10] http://darwin.zoology.gla.ac.uk/~rpage/TreeBASE.

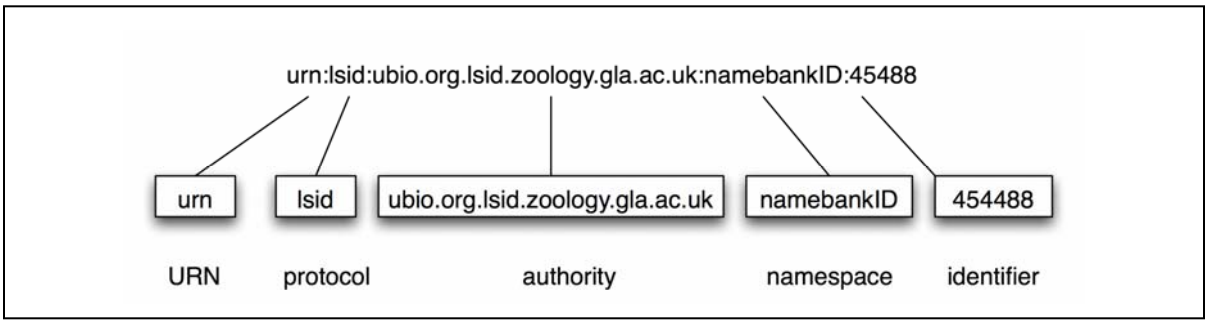

Figure 1. The components of a Life Science Identifier (LSID).

Most readers will be familiar with URLs, which specify the location of a resource the Internet (e.g., http://www.ubio.org/SOAPbrowser/index.php?func=name_detail&ubioID=454488), that is, they "point" to it. They can, in principle serve as a unique identifier, however they are prone to breakage — if the resource being pointed to moves, the URL no longer points to the resource, leading to the dreaded "404 page not found" problem (Dellavalle et al. 2003).

URNs, in contrast, provide a persistent name for a resource, but typically do not provide any information on how to access that resource. A LSID is a Uniform Resource Name (URN). Digital Object Identifiers (DOIs) are another example of a URN, and are widely used in the publishing industry to identify electronic publications. If the resource moves (e.g., one publishing house acquires another, and moves the acquired company's digital resources to a new server) the resource still retains the original DOI. The utility of a URN is somewhat limited, unless there is a mechanism to resolve the URN, that is, to retrieve the named resource. In the case of DOIs, the simplest way to see this mechanism in action is to append a DOI, such as 10.1145/1024694.1024703, to the URL[11] giving in this instance[12], and open the resulting URL in a web browser. In this example the DOI resolves to the electronic version of Herbert et al. (2004).

Life Science Identifiers

Figure 1 shows an example LSID. Each LSID is prefixed by 'urn' indicating that the LSID is a URN, 'lsid' indicates that the identifier is a LSID, then follow the authority, namespace, and identifier components. There may also be an optional revision component to indicate the version of the resource. The authority is a domain name that can be resolved by the Internet DNS (typically a domain name owned by the data provider), the namespace and identifier are specific to the data source which provides the resource. In this case the LSID is a taxonomic name in the uBio database. The authority 'ubio.org.lsid.zoology.gla.ac.uk' is a domain name of a server at the University of Glasgow that serves LSIDs for uBio records. If uBio itself served LSIDs, the domain name could be ubio.org. Note that the uniqueness of the LSID is in part guaranteed by the use of Internet domain names, which are globally unique. Providing that the data source ensures that each combination of namespace and identifier is unique within the data source, the LSID itself will be a globally unique identifier.

A LSID is intended to refer to one unchanging digital object. Hence, if two users retrieve data with the same LSID, they will have the exactly the same data. This contrasts with URLs, where the content may change at any time (for example, if the author of the web page changes the layout). Different versions of a digital object can be identified using the revision part of the LSID. In addition to data there may be metadata associated with a LSID. The LSID standard doesn't require that the metadata remain unchanging.

The Life Science Identifier (LSID) standard specifies a mechanism for resolving a LSID and retrieving the data and/or metadata associated with that LSID. Because a LSID is not a URL, you can't simply paste a LSID into a web browser unless you have additional software installed, such as IBM's LSID Launchpad for Internet Explorer [13] (Figure 2) or the LSID extension for Firefox[14]. The BioPathways Consortium provides a web-based LSID resolver[15]. The LSID shown in (Figure 2) resolves to the IPNI

[11] http://dx.doi.org/.
[12] http://dx.doi.org/10.1145/1024694.1024703.
[13] http://lsid.sourceforge.net/
[14] http://lsid.mozdev.org/
[15] http://lsid.biopathways.org

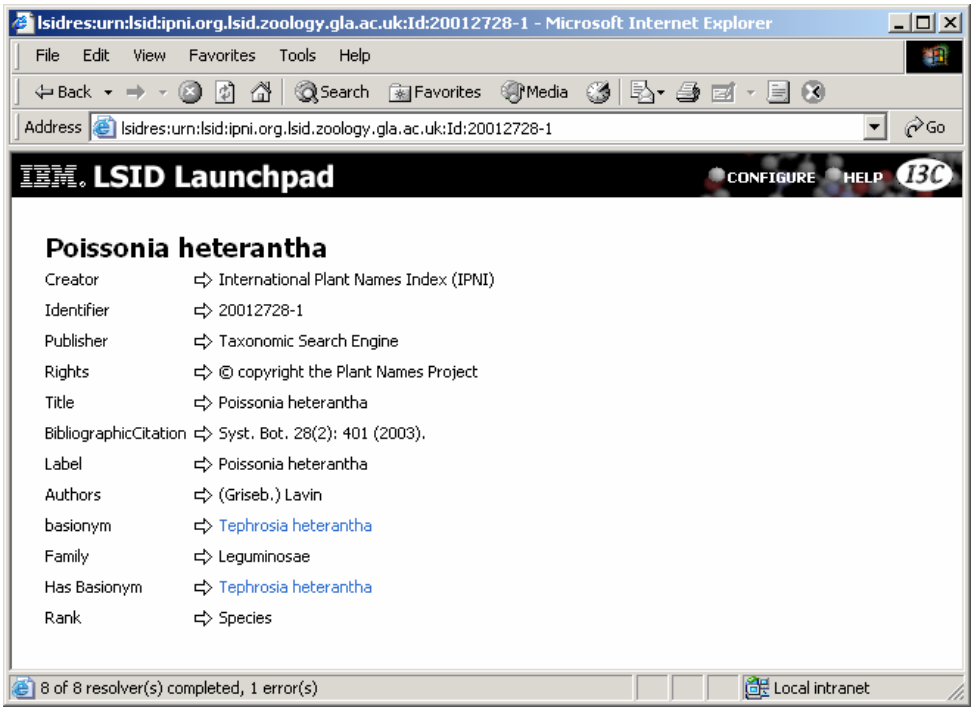

Figure 2. IBM's LSID Launchpad displaying metadata associated with a LSID, this case `urn:lsid:ipni.org.zoology.gla.ac.uk:Id:20012728-1`.

record for the *name Poissionia heterantha* (a plant species). Being a URN, the LSID is a name, not an address.

Currently few data sources serve their own LSIDs, the North Temperate Lakes Long Term Ecological Research project[16] being the notable exception. Third parties, such as the BioPathways Consortium, provide most working LSID authorities.

Search Engine Revisited

The Taxonomic Search Engine (Page 2005) provides LSIDs for each source database that it queries. In order to provide metadata for an LSID, I needed to extract information from each source. One thing which struck me during the development of this search engine was that much of the code I used for the GetDataForID method was being reused when implementing LSID authorities (I use the term "reused" loosely, the code had to be ported from the PHP scripting language to Perl). Hence, if a taxonomic database serves LSIDs, all a taxonomic search engine needs to provide is a means to search those databases (i.e., the NameSearch method).

Metadata

Having recounted how I was drawn to LSIDs in the context of trying to make sense of the taxonomic names in TreeBASE, I will now turn to metadata. The LSID standard doesn't specify that metadata be in any particular format, but the Resource Description Format (RDF[17]) is the most widely used. RDF is a framework for describing relationships between resources, where a resource is connected to another resource by a property. A resource must have a URI. The basic unit in RDF is the *subject*, *property*, *object* triple (Figure 3).

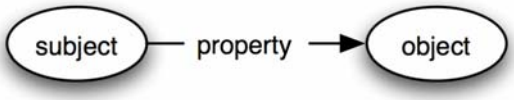

Figure 3. A RDF subject, property, object triple.

[16] http://lsid.limnology.wisc.edu/

[17] http://www.w3.org/RDF/

The object is either a literal string, or another resource. RDF can be represented in a number of ways, most commonly using XML. This is a simple RDF document stating that the World Wide Web Consortium is the publisher of the resource[18].

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
xmlns:rdf =
"http://www.w3.org/1999/02/22-rdf-syntax-
ns#"
xmlns:dc =
"http://purl.org/dc/elements/1.1/"
>
<rdf:Description
rdf:about="http://www.w3.org">
<dc:publisher>World Wide Web
Consortium</dc:publisher>
</rdf:Description>
</rdf:RDF>
```

RDF can be represented in various formats, but XML is most commonly used. The first tag in the document lists the namespaces being used in the document, which ensure that there is no "collision" between terms from different vocabularies, and that there is minimal ambiguity in interpreting a term. For example, `dc:publisher` tells us that the term `publisher` is part of the Dublin Core vocabulary,

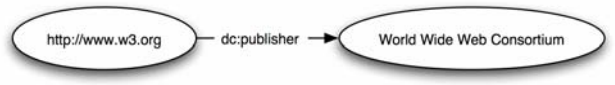

Figure 4. An example RDF statement.

which has the URI[19]. Note that unlike other XML documents, the namespace URI must be retrievable. In the case of Dublin Core, opening the URI in a web browser will return an RDF document describing all the terms in that vocabulary. Namespaces also help make the document more human-readable by defining prefixes that can be used in the document: `rdf:RDF` is more readable than `http://www.w3.org/1999/02/22-rdf-syntax-ns#RDF`. The same information shown in this example can be represented as a graph (Figure 4). Rather than attempt a tutorial on RDF I will illustrate its use with examples. For more background on RDF I recommend Powers (2003).

Why RDF?

It might be tempting to think of RDF as just another variant of XML, and given the substantial investment the taxonomic community has made in developing XML schema for specimens (e.g., ABCD - Access to Biological Collections Data[20], Darwin Core[21], and taxa[22] (Taxonomic Concept Schema) (Kennedy et al. 2005), one could ask why should we contemplate an alternative format? Wang et al. (2005) provide an insightful comparison of XML schema and RDF in the context of 'omic' data standards (e.g., genomics, proteomics. etc.). They conclude that XML schema solve the problem of standardising communication between different data sources at the level of messages, but is not equipped to convey semantics (i.e., meaning). RDF is explicitly designed to model semantics, and I suggest that this is where the real power of LSIDs resides.

Storing and Querying RDF

RDF triples can be stored in a database called a "triple store." A triple store can be thought of as a database with a single table containing multiple rows, each row containing the subject, property, object triple. For this work I use 3store version 2.2.18[23] from the University of Southampton (Harris and Gibbins 2003). 3store uses a MySQL database to physically store the triples, and supports the RDF Data Query Language (RDQL). A more recent version of this software adds support for the SPARQL language.

TAXONOMIC METADATA

This section explores the notion of taxonomic names as metadata. I am not going to attempt to develop an explicit "standard" for taxonomic metadata at this point. Rather, I want to sketch a possible form the metadata could take, then explore what we can do with such metadata. Hence, in the interests of illustrating the ideas I'm going to skirt around the complexities of taxonomic names and concepts (Berendsohn et al. 2003; Kennedy et al. 2005). Figure 5 shows a graph representing the RDF metadata for a single record in the ITIS database for the taxon *Morus bassanus*.

[18] http://www.w3.org/
[19] http://purl.org/dc/elements/1.1/
[20] http://ww3.bgbm.org/abcddocs/
[21] http://darwincore.calacademy.org/
[22] http://www.soc.napier.ac.uk/tdwg/index.php
[23] http://www.aktors.org/technologies/3store/

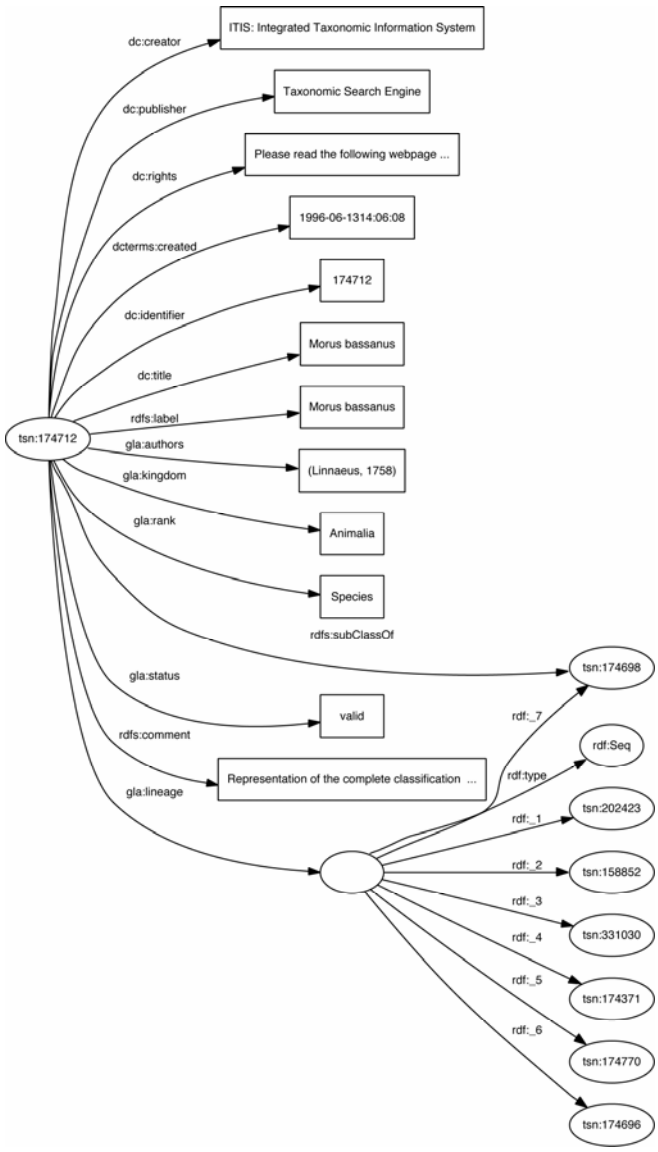

Figure 5. Graph representing the RDF metadata associated with the LSID for the ITIS record for the northern gannet *Morus bassanus*. Boxes represent literals, ovals represent resources (typically identified by their LSID, which to avoid clutter are abbreviated tsn:##### The labelled edges of the graph represent properties, such as the source of the information, the taxonomic name, rank, status, authority, and lineage (represented as a rdf:Seq collection).

Source

Basic information on the source of the data, such as the name of the source database, the original source of the data (e.g., a contributing taxonomic database or specialist), and the date the record was created, can be represented using terms from the Dublin Core.

Copyright

Information on what users can and cannot do with the data can be expressed (typically in free format text) using Dublin Core, or perhaps more efficiently using one of the licenses developed by the Creative Commons[24]. The advantage of the latter is that these licenses are computer-readable, and hence software can discover what it can do with the data without requiring human intervention.

Bibliographic Data

Some taxonomic databases provide bibliographic information. There are various ways this can be represented. The Dublin Core vocabulary provides `dcterms:bibliographicCitation` which can contain the bibliographic information in any suitable format. This is useful where the database stores the bibliographic information as unstructured text (e.g., a single field in the database contains the bibliographic record). If the database stores the bibliographic information in a more structured form, then a more expressive vocabulary such as Publishing Requirements for Industry Standard Metadata (PRISM[25]) could be used. RSS feeds provided by journal publishers make extensive use of this vocabulary (Hammond et al. 2004). It would also be highly desirable to have identifiers for the publication, such as a DOI[26] or a PubMed number.

Person

Although I have not made use of it in this context, the Friend of a Friend project (FOAF[27]) has developed a useful vocabulary for describing people, which could be used to describe authors of scientific names.

Taxonomic Name

A simple approach to representing the taxonomic name itself is through use of the `dc:title` and `rdfs:label` properties. One reason for adopting these tags is for consistency with tools such as LSID Launchpad (Figure 2), which use these tags to display a title for LSID metadata.

For some information about taxonomic names, such as the authors of the name, the taxonomic rank,

[24] http://creativecommons.org/
[25] http://www.prismstandard.org/
[26] http://www.doi.org/
[27] http://www.doi.org/

and synonymy, we need a vocabulary that is specific to biological taxonomy. I will use a simple vocabulary that describes the bare minimum of terms need. In the examples below I will use the namespace prefix `gla`, which is short for `urn:lsid:lsid.zoology.gla.ac.uk:predicates`, such that each property is defined by a LSID. This means that for any property there is associated metadata that describes that property.

Relationships Among Names

Given that each name has an LSID, we can describe the relationship between two names using the appropriate property. There are various kinds of relationships among taxonomic names that we might wish to model, of which synonymy is perhaps the most important. There are various kinds of synonyms, which we can loosely characterise as "objective" and "subjective." Two names are objective synonyms if there is no doubt that the two names refer to the same taxon. For example, if a species moves from one genus to another a new name combination results. These two names clearly refer to the same entity — they have the same type specimen. In other cases whether the name refers to the same entity may be contentious. Consider the case where there are two species names, each with different holotypes. One taxonomic authority may regard these species as distinct, a second authority may regard them as the same species, and hence treat the names as synonyms. In the later case, this is an inference based (ideally) on data, not a simple consequence of the appropriate rules of nomenclature.

As an example of objective synonymy, the plant originally described as *Tephrosia heterantha* Griseb. has been placed in the genus *Poissonia heterantha* by Lavin et al. (2002). Hence, *Tephrosia heterantha* is the basionym of *Poissonia heterantha*. We can represent this relationship in RDF like this:

```
<rdf:Description
rdf:about="urn:lsid:ipni.org...:Id:200127
28-1">
   <dc:title>Poissonia
   heterantha</dc:title>
   <gla:hasBasionym
   rdf:resource="urn:lsid:ipni.org...:Id:
   520610-1"/>
</rdf:Description>
```

where the names *Poissonia heterantha* and *Tephrosia heterantha* have the LSIDs

`urn:lsid:ipni.org.lsid.zoology.gla.ac.uk:Id:20012728-1`

and

`urn:lsid:ipni.org.lsid.zoology.gla.ac.uk:Id:520610-1,`

respectively. These LSIDs are generated by the Taxonomic Search Engine (Page 2005) for data in the IPNI database. The inverse relationship is `gla:isBasionymOf`. According to the IPNI database, *Tephrosia heterantha* is also the basionym for *Coursetia heterantha* (Figure 6), which we can represent like this:

```
<rdf:Description
rdf:about="urn:lsid:ipni.org...:Id:520610-
1">
     <dc:title>Tephrosia
     heterantha</dc:title>
     <gla:isBasionymOf
     rdf:resource="urn:lsid:ipni.org...:Id:9
     44651-1"/>
    <gla:isBasionymOf
    rdf:resource="urn:lsid:ipni.org...:Id:20
    012728-1"/> </rdf:Description>
```

Figure 6. Two names linked to their basionym. *Poissonia heterantha* and *Coursetia heterantha* are synonyms because they share the same basionym, *Tephrosia heterantha*.

Kinds of Synonyms

The depth of information on synonymy varies across taxonomic databases, and even within databases. For example, IPNI comprises three source databases: Index Kewensis, the Gray Card Index, and the Australian Plant Names Index. The Gray Card Index specifies whether two names are "nomenclatural synonyms," but does not for example specify the exact nature of the relationship (i.e., whether one name is a basionym of the other). Based on the kinds of synonyms reported by the data sources queried by the Taxonomic Source Engine, we can construct a graph depicting the relationships between synonym types (Figure 7). These relationships are modelled by a

simple RDF Schema (RDFS[28]). The property `gla:synonym` has the sub-properties `gla:objectiveSynonym` and `gla:hasAccepted-Name`. The later is based on the term used in ITIS to specify which name that database regards as the correct name to use for a taxon. The accepted name may in fact be an objective synonym, but because ITIS does not provide sufficient information to determine that, I've chosen not to regard this as an objective synonym. If a database merely specifies that a name is a synonym then the `gla:synonym` property can be used. The following is an abbreviated version of this schema:

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/2
    2-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/
    1.1/"
    xmlns:rdfs="http://www.w3.org/2000/01/
    rdf-schema#"
>

<rdf:Property
rdf:about="urn:lsid:...:predicates:synony
m">
    <dc:title>Synonym</dc:title>
    <rdfs:range
    rdf:resource="http://www.w3.org/2000/0
    1/rdf-schema#Resource" />
</rdf:Property>

<rdf:Property
rdf:about="urn:lsid:...:predicates:object
iveSynonym">
    <rdfs:subPropertyOf
    rdf:resource="urn:lsid:...:predicates:
    synonym"/>
    <dc:title>Objective Synonym</dc:title>
    <rdfs:range
    rdf:resource="http://www.w3.org/2000/0
    1/rdf-schema#Resource" />
 </rdf:Property>

<rdf:Property
rdf:about="urn:lsid:...:predicates:isBasi
onymOf">
  <rdfs:subPropertyOf
  rdf:resource="urn:lsid:...:predicates:o
  bjectiveSynonym"/>
  <dc:title>Basionym Of</dc:title>
  <rdfs:range
  rdf:resource="http://www.w3.org/2000/01
  /rdf-schema#Resource" />
</rdf:Property>
</rdf:RDF>
```

In this schema synonyms are modelled using the `rdf:Property` property, and if one kind of synonym is a refinement of another we use the `rdsf:subPropertyOf` property. Hence, the `isBasionymOf` property is a refinement of the `objectiveSynonym` property. If we ask for the objective synonyms of a name, we should recover any synonym that either has the property `objectiveSynonym`, or a property that is a subproperty of `objectiveSynonym`. As an example, consider the taxon names shown in Figure 6 and the corresponding RDF:

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
    xmlns:dc="http://purl.org/dc/elements/
1.1/"
    xmlns:gla="urn:lsid:lsid.zoology.gla.a
    c.uk:predicates:"
 >

<rdf:Description
rdf:about="urn:lsid:ipni.org...Id:2001272
8-1">
   <dc:title>Poissonia h
   eterantha</dc:title>
   <gla:hasBasionym
   rdf:resource="urn:lsid:ipni.org...Id:5
   20610-1"/>
</rdf:Description>

<rdf:Description
rdf:about="urn:lsid:ipni.org...:Id:944651
-1">
   <dc:title>Coursetia
   heterantha</dc:title>
   <gla:hasBasionym
   rdf:resource="urn:lsid:ipni.org...:Id:
   520610-1"/>
 </rdf:Description>

<rdf:Description
rdf:about="urn:lsid:ipni.org...:Id:520610
-1">
    <dc:title>Tephrosia
    heterantha</dc:title>
   <gla:isBasionymOf
   rdf:resource="urn:lsid:ipni.org...:Id:
   944651-1"/>
   <gla:isBasionymOf
   rdf:resource="urn:lsid:ipni.org...:Id:
   20012728-1"/>
</rdf:Description>
</rdf:RDF>
```

[28] http://www.w3.org/TR/rdf-mt/

The following RDQL query asks for the objective synonyms of *Tephrosia heterantha* (`urn:lsid:ipni.org.lsid.zoology.gla.ac.uk:Id:520610-1`):

```
SELECT ?synonym
WHERE
(<urn:lsid:ipni.org.lsid.zoology.gla.ac.uk:Id:520610-1>,
<gla:objectiveSynonym>, ?id),
(?id, <dc:title>, ?synonym)
USING gla for
<urn:lsid:lsid.zoology.gla.ac.uk:predicates:>,
Dc for <http://purl.org/dc/elements/1.1/>
```

This query returns the result

```
 ?synonym
"Poissonia heterantha"
"Coursetia heterantha"
```

Note that our RDQL query doesn't mention the property `isBasionymOf` by name, but it returns the two names for which *Tephrosia heterantha* is the basionym because `isBasionymOf` property is a refinement of the `objectiveSynonym` property. Put another way, if a name is an `isBasionymOf` of another name, this entails that the name is also an `objectiveSynonym`. Unfortunately, support for entailment in RDF query languages is currently patchy (Broekstra 2005). Given the RDF schema above we can see that if a name *a* is a basionym of name *b*, then this entails the following statements:

1. *a* is an objective synonym of *b*
2. *a* is a synonym of *b*

The RDQL engine that comes with version 2.2.18 of 3store will infer the first statement, but not the second. As a (I hope) temporary fix until support for entailment improves, we can add additional edges to the graph to ensure that RDQL can also infer statement 2.

Inferring Synonymy

Once we start modelling relationships between names using graphs (which comes naturally if we adopt RDF) then we can quickly discover limitations in the information provided by taxonomic databases. For example, in the case of *Poissonia heterantha* and *Coursetia heterantha* shown in Figure 6, there is no direct link between these two names (in graph terminology, there are no edges connecting the two nodes). This is because the IPNI records for *Poissonia heterantha* and *Coursetia heterantha* do not make explicit that these two names are synonyms — the user has to work this out from the fact that both names share the same basionym. We can make this relationship explicit by computing the transitive closure of the graph. The transitive closure of a graph is obtained by adding an edge between nodes *i* and *j* if node *j* is reachable from node *i*, that is, there is a path from node *i* and node *j*. Figure 8a shows the original graph representing the relationships between *Poissonia heterantha*, *Coursetia heterantha*, and *Tephrosia heterantha*, and the transitive closure of that graph (Figure 8b), showing that Poissonia heterantha and *Coursetia heterantha* are synonyms.

Vernacular Names

Vernacular or common names exist for many taxa, and in many languages. We can represent these using a `vernacularName` tag, and use the `xml:lang` attribute to specify the language.

```
<vernacularName xml:lang="en">Common Swift</
vernacularName>
```

Parts of Names

For names that have more than one part there is a logical relationship between those parts. For example, the trinomial *Coursetia caribaea caribaea* is a variety of the species *Coursetia caribaea*, which in turn is a species of the genus *Coursetia*. Some database models (Pyle 2004) store the individual components of a multinomial name separately to facilitate tracking name changes, and searches for epithets. Under such a model, the variety *caribaea* has as its parent the species *caribaea*, which in turn is a child of the genus *Coursetia*.

Although this relationship also expresses a classification (i.e., that this species belongs in the genus *Coursetia*), it is useful to separate the notion of a logical relationship between the parts of a name (a species name must include both genus name and specific epithet) and a classification. Typically in a classification only the accepted names are part of the parent-child hierarchy. The basionym for *Coursetia caribaea* is *Galega caribaea*, hence in this instance *caribaea* is a child of *Galega*. Even though *Galega caribaea* is not the accepted name for this plant, the

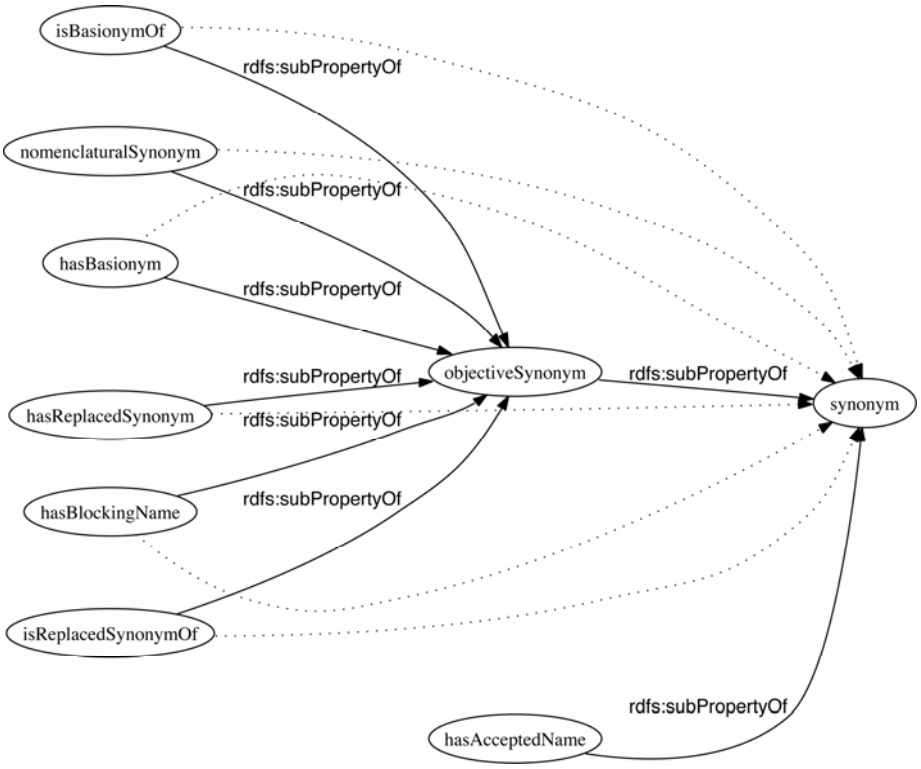

Figure 7. Examples of different types of synonym. The property `rdfs:subPropertyOf` links a more restricted kind of synonym to a more general kind of synonym. Dotted lines indicate edges that need to ensure that RDQL infers all the relationships entailed by the property `rdfs:subPropertyOf` (see the text).

logical relationship between the two components of the name remains. We can model this relationship using the `dcterms:isPartOf` and `dcterms:hasPart` properties from the Dublin Core Terms (Figure 9).

Classification

A classification of a set of taxa can be regarded as a rooted tree with nodes are taxa and are labelled with the accepted names of those taxa. This, for example, is the model used by ITIS. In the case of taxa below the rank of genus, for the accepted name the classification will mirror the `dcterms:isPartOf` and `dcterms:hasPart` relationships among the components of the name.

It is tempting to model this taxonomic hierarchy using the `rdfs:subClassOf` property, particularly as `rdfs:subClassOf` is transitive. If each taxon in a classification knows its parent taxon, then we could traverse the path from the tip to the root of a classification by using `rdfs:subClassOf`. However, consider what happens if we want to go in the other direction, for example, if we want to find all birds. We would need to go through each node in the classification and go down the path to the root until we either (a) find "Aves" or (b) reach the root of the tree. We can speed things up by storing for each node the path from that node to the root of the tree. This path is the node's lineage. One natural way to do this would be to use the `rdf:Seq` class, which stores an ordered sequence of URIs. For example, the lineage for *Morus bassanus* in ITIS is:

```
<gla:lineage>
<rdf:Seq>
<rdf:li rdf:resource =
"urn:lsid:itis.usda.gov...:tsn:202423"/>
<rdf:li rdf:resource =
"urn:lsid:itis.usda.gov...:tsn:158852"/>
<rdf:li rdf:resource =
"urn:lsid:itis.usda.gov...:tsn:331030"/>
<rdf:li rdf:resource =
"urn:lsid:itis.usda.gov...:tsn:174371"/>
<rdf:li rdf:resource =
"urn:lsid:itis.usda.gov...:tsn:174770"/>
<rdf:li rdf:resource =
"urn:lsid:itis.usda.gov...:tsn:174696"/>
```

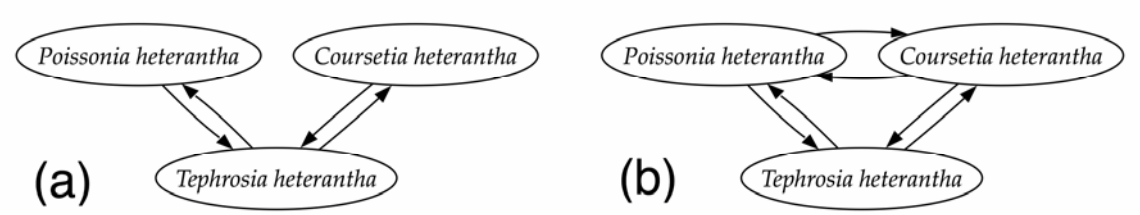

Figure 8. (a) The relationships between three names for the same plant (Figure 7), and (b) the transitive closure of the graph. An edge between two nodes indicates that the corresponding names are synonyms.

```
<rdf:li rdf:resource =
"urn:lsid:itis.usda.gov...:tsn:174698"/>
</rdf:Seq>
</gla:lineage>
```

We use a sequence because order is important (in this example the order is from higher to lower taxon).

By storing the lineage for a node, we can find all members of a given higher taxon by finding those that have the corresponding URI in their lineage. Given than both approaches to representing a classification have their merits, here I use both: `rdfs:subClassOf` to specify the immediate parent, and `rdf:Seq` to specify the complete lineage.

Links Between Data Sources

Some databases store links to information associated with the same name in other databases. Examples include NCBI LinkOut, which links a taxon in GenBank to various external databases, and uBio, which records the source of names it stores in its data warehouse. These links can be modelled using the `rdfs:sameAs` property. This (and other) properties can be used to represent relationship between names in different databases, but also other data, such as specimens, sequences, and publications. For example, metadata for a specimen could link the name assigned to that specimen to the record for that name in uBio.

TREEBASE (AGAIN)

Federated search engines, globally unique identifiers, metadata, and the semantic web seem somewhat removed from the original problem, namely making sense of TreeBASE. Let me work through an example of how I think these technologies can help. The ultimate goal is to be able to query the TreeBASE database with a taxonomic name, and recover all the studies that contain that taxon. There are some significant obstacles in our way. In many cases, the names in TreeBASE (stored in a field called `TaxonName`) are not names of taxa but names of *Operational Taxonomic Units* (OTUs). That is, the names might be a combination of taxon name and some other identifier, such as a specimen or voucher code, or a GenBank accession number. For example, in TreeBASE Study S813 (Lavin et al. 2002) we have TaxonName records such as "Poissonia heterantha 5832", "Poissonia heterantha 5862", and so on. Another obstacle is that TreeBASE does not have any notion of synonymy, so that if two studies refer to the same taxon using different names, TreeBASE cannot tell the user that the two taxa are, in fact, the same. One solution would be to map each taxon name in TreeBASE to one or more external data sources, and to use information from that external data source — such as lists of synonyms — to improve the performance of our query.

Name Changes in Legumes

As our phylogenetic knowledge of a group of organisms grows it is not uncommon for this new understanding to be reflected in taxonomic changes. Consequently, names used for the same taxon in successive studies submitted to TreeBASE may have changed. As a concrete example, different names have been used for same legume taxa in successive papers published by Matt Lavin and collaborators. Hence, TreeBASE study S754 (Lavin et al. 2001) includes names such as *Coursetia heterantha* and *C. weberbaueri*. However, in Study S813 (Lavin et al. 2002) these taxa were moved to the genus *Poissonia*. Ideally, if we search TreeBASE for "Coursetia heterantha" we should find both study S754 and S813, because both studies contain this taxon, albeit under different names. One way to add this "taxonomic intelligence" to TreeBASE would be to map names in that database to external name sources that knew the relationship between the different names.

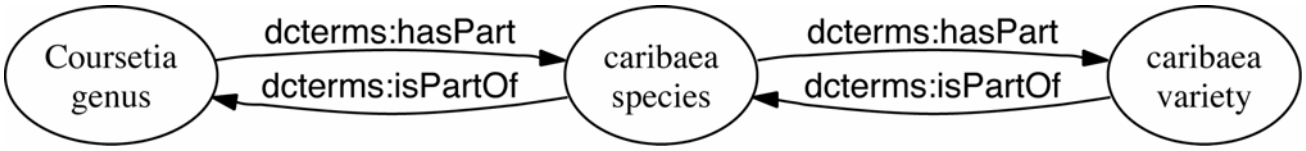

Figure 9. An example of the relationships between the component parts of the name *Coursetia caribaea caribaea*, modelled using the `dcterms:isPartOf` and `dcterms:hasPart` properties.

Mapping Names

To create the mapping I did the following. Each TreeBASE TaxonName for study S813 was "cleaned" to remove extraneous suffixes indicating voucher codes, etc., using uBio's findIT web service. The resulting cleaned names were then submitted to IPNI using to the Taxonomic Search Engine web service (Page 2005). For each IPNI name the corresponding LSID was resolved and the metadata obtained for that LSID (and for any LSID referred to by the metadata).

At each step we incrementally gain some knowledge, which can be expressed in RDF and added to a triple store. For, example, we can represent what TreeBASE knows about a `TaxonName` in RDF:

```
<rdf:Description
rdf:about="urn:lsid:treebase.org...:Taxon
ID:T30615">
<dc:title>Poissonia heterantha
5832</dc:title>
<dc:identifier>T30615</dc:identifier>
</rdf:Description>
```

This simply asserts that there is a taxon with the identifier "T30615" and the name "Poissonia heterantha 5832". Once we have cleaned the name and mapped it to IPNI we can add a statement asserting that TreeBASE `TaxonID` T30615 is the same as IPNI record 20012728-1:

```
<rdf:Description
rdf:about="urn:lsid:treebase.org...:TaxonID:
T30615">
<rdfs:sameAs rdf:ID="T83"

rdf:resource="urn:lsid:ipni.org...:Id:200127
28-1" />
</rdf:Description>
```

The attribute `rdf:ID`="T83" is explained below in the section on reification. The final information we can add to our triple store is the metadata for the IPNI record (shown here in abbreviated form):

```
<rdf:Description
rdf:about="urn:lsid:ipni.org...:Id:200127
28-1">
<dc:title>Poissonia heterantha</dc:title>
<gla:hasBasionym
rdf:resource="urn:lsid:ipni.org...:Id:520
610-1"/>
</rdf:Description>
```

This states that the taxon's name is "Poissonia heterantha", and its basionym is `urn:lsid:ipni.org.lsid.zoology.gla.ac.uk:Id:520610-1`.

Queries

Once we have mapped each TreeBASE `TaxonName`, we can now search for TreeBASE taxa using a proper scientific name, e.g.:

```
SELECT ?TaxonID, ?TaxonName
WHERE (?ipni, <dc:title>, "Poissonia
heterantha")
      (?tb, <rdfs:sameAs>, ?ipni )
      (?tb, <dc:title>, ?TaxonName )
      (?tb, <dc:identifier>, ?TaxonID )
USING dc FOR
<http://purl.org/dc/elements/1.1/>,
rdfs FOR <http://www.w3.org/2000/01/rdf-
schema#>
```

This query finds the IPNI record for the name "Poissonia heterantha", then finds all TreeBASE taxa that are mapped to that IPNI record and lists their `TaxonID` and *TaxonName*. For study S813 this query yields:

```
?TaxonID     ?TaxonName
"T30610"      "Poissonia heterantha"
```

```
"T30615"      "Poissonia heterantha 5832"
"T30616"      "Poissonia heterantha 5862"
"T30714"      "Poissonia heterantha 5856"
"T30590"      "Poissonia heterantha 5843"
"T30715"      "Poissonia heterantha 5800"
"T30713"      "Poissonia heterantha 5860"
"T30589"      "Poissonia heterantha 5785"
```

Query Expansion and Taxonomic Intelligence

Simply finding taxa by name doesn't add much to TreeBASE, and indeed we could retrieve all variants of "Poissonia heterantha" directly via the web interface to TreeBASE by using the search term "Poissonia heterantha@". However, the metadata for the IPNI records enables us to query by synonyms. For example, we know that *Coursetia heterantha* is a synonym of *Poissonia heterantha* (Figure 6), hence we should be able to use "Coursetia heterantha" as a search term and retrieve records for *Poissonia heterantha*, as in the following RDQL query:

```
SELECT ?TaxonID, ?TaxonName
   WHERE (?ipni, <dc:title>, "Coursetia
   heterantha")
    (?ipni,<gla:hasBasionym>,?basionym)
    (?basionym,<gla:isBasionymOf>,?synonym)
    (?tb, <rdfs:sameAs>, ?synonym )
    (?tb, <dc:title>, ?TaxonName )
    (?tb, <dc:identifier>, ?TaxonID )
   USING dc FOR
   <http://purl.org/dc/elements/1.1/>
     gla for
     <urn:lsid:lsid.zoology.gla.ac.uk:predic
     ates:>
   rdfs FOR <http://www.w3.org/2000/01/rdf-
   schema#>
```

For study S813 this query yields:

```
?TaxonID        ?TaxonName
"T30610"        "Poissonia heterantha"
"T30615"        "Poissonia heterantha 5832"
"T30616"        "Poissonia heterantha 5862"
"T30714"        "Poissonia heterantha 5856"
"T30590"        "Poissonia heterantha 5843"
"T30715"        "Poissonia heterantha 5800"
"T30713"        "Poissonia heterantha 5860"
"T30589"        "Poissonia heterantha 5785"
```

This result is the same as if we had searched for "Poissonia heterantha". This query is more complex than it needs to be because, as discussed above (see Figure 8), IPNI does not explicitly state that *Poissonia heterantha* and *Coursetia heterantha* are synonyms. Consequently, in the metadata retrieved from IPNI there is no triple where the subject is *Poissonia heterantha* and the object is *Coursetia heterantha* (or visa versa). However, the key point is that by combining metadata from TreeBASE and IPNI, we have added the ability to handle synonyms to TreeBASE.

Reification

Although sometimes called the "big ugly of RDF (Powers 2003), reification is a useful technique which enables us to make statements about statements. For example, when mapping names in TreeBASE onto names in an external database, some names have an exact match (such as `TaxonID` T30610 "Poissonia heterantha"), and other names match once they have been cleaned (for example, `TaxonID` T30615 "Poissonia heterantha 5832"). It would be useful to be able to state what type of match was obtained for each name, and reification provides a mechanism for doing so. Consider this snippet of RDF:

```
<rdf:Description
rdf:about="urn:lsid:treebase.org...:Taxon
ID:T30615">
<rdfs:sameAs rdf:ID="T83"
rdf:resource="urn:lsid:ipni.org...:Id:200
12728-1" />
</rdf:Description>
```

which states that TreeBASE `TaxonID` T30610 is the same as IPNI record 20012728-1. The `rdfs:sameAs` statement has a `rdf:ID` attribute with the value "T83". This enables us to refer to this statement and hence provide details what we mean by "same as", for example:

```
<rdf:Description rdf:about="#T83">
<dc:description>approximate
</dc:description>
</rdf:Description>
```

This RDF triple informs us that statement T83 is an approximate match (i.e., the string "Poissonia heterantha 5832" is not an exact match for the string "Poissonia heterantha"). Note that we could add additional statements, for example describing the degree to which the two strings differ, a date and time stamp for when the link between the two names was made, and any comments on the link.

It might seem simpler to represent the kind of match like this (i.e., no reification):

```
<rdf:Description
rdf:about="urn:lsid:treebase.org...:Taxon
ID:T30615">
<rdfs:sameAs
rdf:resource="urn:lsid:ipni.org...:Id:200
12728-1" />
<dc:description>approximate
</dc:description>
</rdf:Description>
```

However, in this case the `dc:description` property refers to the TreeBASE taxon T30615, and not the statement that links this taxon with the IPNI record.

DISCUSSION

My intention here has been to sketch out the case for using RDF and LSIDs to model taxonomic names. In one sense these two technologies are complementary, although strictly speaking neither requires the other. RDF requires URIs for resources, and LSIDs are a natural candidate for these URIs, but some other URI could be used. LSIDs have associated metadata for which RDF is the obvious candidate, but other formats could be used. However, I think these two technologies have the advantage of being available, relatively well understood, and in the case of RDF, part of a much broader effort (the semantic web). RDF is a relatively simple format, but the existence of numerous vocabularies that are relevant to biodiversity informatics, and its support for inference makes it potentially very powerful. It also introduces the notion of modelling relationships between taxonomic names using graphs, which can yield new insights (such as computing all synonyms using transitive closure).

I am fully aware that the relationships between taxonomic names are more complicated that I've sketched here (Berendsohn et al. 2003; Kennedy et al. 2005). I have deliberately tried to keep the RDF described here as simple as possible, in the belief that keeping things simple and maximising the use of existing vocabularies is more productive than trying to design complex, domain specific schema. However, one area that could be usefully explored is the use of ontologies to explicitly model relationships between properties and to provide better support for inference. We have also seen that playing with even simple examples illustrates the potential to make inferences from very simple metadata, and also that existing taxonomic databases lack taxonomic intelligence, that is, they don't always know what they know.

Future

If we adopt RDF for modelling taxonomic names and other objects in biodiversity informatics, then this leads naturally to rethinking the way we might develop biodiversity databases. Most work in this area concentrates on using relational databases to store data (Morris 2005) and XML schema for exchanging data (e.g., ABCD, Darwin Core, and Taxonomic Concept Schema) (Kennedy et al. 2005). Both these technologies have a role to play. Relational databases support data integrity and a sophisticated query language (SQL), however they have limitations — database schema can rapidly become large, complex, and domain specific. Furthermore, the emphasis in designing such schema is on internal data integrity, rather than relationships with external data sources. This is a major limitation in an environment where most data is stored elsewhere. XML schema are good at describing messages, but poor at communicating meaning (Wang et al. 2005). Like relational database schema, XML schema can rapidly become large and unwieldy.

A different (but complementary) vision is of a triple store containing RDF triples, where a globally unique identifier explicitly identifies every resource. Where possible, identifiers can be resolved to a source of metadata, itself in RDF format. As shown in the TreeBASE example above, one can rapidly construct and populate a triple store which supports inference of the sort that is relevant to the question at hand. Simply specifying a relationship between two names, and adding the metadata for those names enables us to infer relationships between different data elements in that database. I am not suggesting that RDF and triple stores are the only technology we should use — relational databases and XML schema have important roles to play. Rather, I suggest that RDF and triple stores are ideally suited to support a distributed query system of the kind that biodiversity informatics aspires to provide.

Resources

The RDF examples presented here (often somewhat abbreviated) are available online[29].

[29] http://darwin.zoology.gla.ac.uk/~rpage/lsid/examples

ACKNOWLEDGEMENTS

Part of this work was funded by BBSRC grant BB/C004310/1. I thank the reviewers (Walter Berendsohn, Anton Güntsch, Markus Döring, and one who remained anonymous) for their helpful comments.

LITERATURE CITED

Berendsohn, W. G., M. Döring, M. Geoffroy, K. Glück, A. Güntsch, A. Hahn, R. Jahn, W.-H. Kusper, J. Li, D. Röpert, and F. Specht. 2003. MoreTax: handling factual information linked to taxonomic concepts in biology. Bundesamt für Naturshutz, Bonn.

Broekstra, J. 2005. Storage, querying and inferencing for semantic web languages. PhD thesis. Vrije Universiteit.

Clark, T., S. Martin, and T. Liefeld. 2004. Globally distributed object identification for biological knowledgebases. Briefings in Bioinformatics 50:59-70.

Dellavalle, R. P., E. J. Hester, L. F. Heilig, A. L. Drake, J. W. Kuntzman, M. Graber, and L. M. Schilling. 2003. Going, going, gone: lost Internet references. Science 302:787-788.

Hammond, T., T. Hannay, and B. Lu. 2004. The role of RSS in science publishing: syndication and annotation on the Web. SIGMOD Rec. 10.

Harris, S., and N. Gibbins. 2003. 3store: Efficient bulk RDF storage. Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems (PSSS'03), Sanibel Island, Florida, 1-15.

Herbert, K. G., N. H. Gehani, W. H. Piel, J. T. Wang, and C. H. Wu. 2004. BIO-AJAX: an extensible framework for biological data cleaning. SIGMOD Rec. 33:51-57.

Kennedy, J. 2003. Supporting taxonomic names in cell and molecular biology databases. OMICS: A Journal of Integrative Biology 7:13-16.

Kennedy, J., R. Kukla, and T. Paterson. 2005. Scientific names are ambiguous as identifiers for biological taxa: their context and definition are required for accurate data integration. Pp. 80-95 in B. Ludäscher and L. Raschid, eds. Data Integration in the Life Sciences: Second International Workshop, DILS 2005, San Diego, CA, USA, July 20-22, 2005 (Lecture Notes in Computer Science 3615). Springer Verlag.

Lavin, M., M. F. Wojciechowski, P. Gasson, C. E. Hughes, and E. Wheeler. 2002. Phylogeny of robinioid legumes (Fabaceae) revisited: *Coursetia* and *Gliricidia* recircumscribed, and a biogeographical appraisal of the Caribbean endemics. Systematic Botany 28:387-409.

Lavin, M., M. F. Wojciechowski, A. Richman, J. Rotella, M. J. Sanderson, and A. Beyra-Matos. 2001. Identifying Tertiary radiations of Fabaceae in the Greater Antilles: alternatives to cladistic vicariance analysis. International Journal of Plant Science 162:S53-S76.

Morris, P. J. 2005. Relational database design and implementation for biodiversity informatics. Phyloinformatics 7:2.

Page, R. D. M. 2004. Phyloinformatics: towards a phylogenetic database. Pp. 219-241 in J. T. L. Wang, M. J. Zaki, H. T. T. Toivonen and D. Shasha, eds. Data Mining in Bioinformatics. Springer Verlag.

Page, R. D. M. 2005. A taxonomic search engine: federating taxonomic databases using web services. BMC Bioinformatics 6:48.

Piel, W. H., M. J. Donoghue, and M. J. Sanderson. 2002. TreeBASE: a database of phylogenetic knowledge. Pp. 41-47 in J. Shimura, K. L. Wilson and D. Gordon, eds. To the interoperable "Catalog of Life" — with partners Species 2000 Asia Oceanea. Research Report from the National Institute for Environmental Studies No. 171, Tsukuba.

Powers, S. 2003. Practical RDF. O'Reilly, Sebastopol, California.

Pyle, R. L. 2004. Taxonomer: a relational data model for managing information relevant to taxonomic research. Phyloinformatics 1.

Stein, L. 2003. Integrating biological databases. Nature Reviews: Genetics 4:337-345.

Wang, X., R. Gorlitsky, and J. S. Almeida. 2005. From XML to RDF: how semantic web technologies will change the design of 'omic' standards. Nature Biotechnology 23:1099-1103.